DIGITGRAPH
P. O. Box 4249
Tucson, Arizona 85717

RSTS-11 SIG DECUS

MAY 1977 RSTS-11 SPECIAL INTEREST GROUP NEWSLETTER VOL 4 NO 2

FROM THE EDITOR

A lot of water has gone under the bridge since the last newsletter - some good and some polluted.

On the good side, the meeting in Chicago to discuss V6B was super. It was attended by over 200 RSTS users. The effort by Stu Katz to make the meeting happen and work like clockwork with just a couple of months to prepare was just incredible. Many other people need to be thanked, but let me just mention a few of them. Maryann Oskirko of DECUS took time from her alreacy overloaded schedule to make the arrangements with the hotel and the registration fcr the meeting. Dian Pekin and her development people at DEC also took time above and beyond the call of duty to prepare and attend the meeting. Also, the DECUS US Board has graciously allowed us to use some of the income frcm the Chicago meeting to send the newsletter first class.

On the good side as a whole is V6B. I've been running it since the end of February and wouldn't want to go back to V6A. The new sysgen procedures are sure nice and patching is easier, thank goodness. The main things that I've enjoyed are the long statements, long variable names, and echo control. I wish we could get the file structure under RSTS redone now, some more things done to CUSPS, and the true ability to handle remote printers; then my personal WISH list for RSTS would be filled.

On the good side, the Spring DECUS Symposium from May 31 to June 3 has more RSTS sessions than ever. We have reached the pcint where we now are forced to schedule parallel RSTS sessions. Bill Sconce took on the impossible task of scheduling the RSTS sessions this time and did a much better job than the guy who did it in the past.

On the not so good side, the reason that this newsletter took so long to get out was that the last newsletter was censored for having a notice of a user with some used RP03 disks for sale. As far as I can tell the last newsletter was held up at least two weeks. During the past couple of months I have been rattling chains and making loud noises to get some policies straightened out. There is now a set policy for the review of newsletters and I am assured that there will no longer be any long delays caused by reviews of newsletters.

On the not so good side, DEC still has not provided any relief for the problems of getting user modified CUSPS and packages like STAT 11 in the DECUS library. Worst of all TECO, which is needed more than ever with the new longer statements is still trapped in a bureaucratic mess. It is time for DEC to listen to its users, not its lawyers.

Bob Branton
3115 Brincrest Dr.
Dallas, Texas 75234
(214) 243-3451
(214) 243-0420 (after 5)

Here is a list of the hard working people who are chairing RSTS Subgroups:

A. COMMERCIAL RSTS USERS
Stu Katz
Commercial Computer Services
401 Fullerton Parkway
Chicago, Ill. 60614
(312) 929-7400

EDITOR OF COMMERCIAL SECTION
Murry Stinson
Scientific Placement
2700 South Post Oak Road
Suite 2430
Houston, Tx 77056
(713) 621-9811

EDUCATIONAL RSTS USERS
Howard Kipple
New Mexico Junior College
Hobbs, New Mexico 88240
(505) 392-6526

LARGE RSTS SYSTEMS GROUP
Joan Salkin
Crown Zellerbach
1 Bush Street
San Francisco, California 94119
(415) 823-5504

CUSP EDITOR
Jon Spencer
PASCAL Computing
432 S. Barrington Ave.
Los Angeles, Ca. 90049

TECHNICAL EDITOR
Carl Marbach
Acme-Hardesty Company
910 Fox Pavilion
Jenkintown, Pa. 19046
(215) 886-0200

JSTATS, DSTATS DOCUMENTATION
Mike Mayfield
Digital Equipment Corporation
871 South Nash St.
El Segundo, Ca. 90245
(213) 640-1830

B. WORKING GROUPS:

RSTS USER PROFILES
Bill Tabor
Florida Computer, Inc.
99 NW 183rd St.
N. Miama, Florida 33169
(305) 652-1710

PROGRAMMING STANDARDS DEVELOPMENT
Roy Swatzell, Jr.
V.A. Hospital
Room 2 R 12
Birmingham, Al 35233
(205) 933-8103 Ext. 239

RSTS DECUS LIBRARY PROGRAM
Ardoth Wilson
Central State University
100 N. University
Edmond, Ok. 73034
(405) 341-2980 Ext. 321

RSTS User WISH List Development
Bill Sconce
Industrial Specialties
PO Box 4228
Louisville, Ky 40204
(502) 895-2910

ADDITIONS TO SPRING DECUS

Two RSTS sessions have been added to the Spring DECUS schedule. They both are very excellent topics and I know you will want to mark them in your schedule.

Performance Evaluation of RSTS

Performance Evaluation will be a discussion to RSTS system performance from the viewpoint of the user. Al Saloky of DEC, who probably has done more research into RSTS performance than anyone, will present this session. Al will discuss software and hardware montiorins alons with simulation techniques he has developed for RSTS. He will so into the interpetation of the resultant statistics and how they can be used to enhance the performance of a RSTS system. He will also discuss the effect of networks on RSTS performance if time permits. The session takes the place of the BASIC+ programmins techniques session which has been cancelled. It is scheduled for:

Friday June 3
8:30 - 9:30
Independence Room

LASER: Lawrence Accountins System Extensions for RSTS/E

LASER consists of modifications to several DIGITAL CUSPs and the addition of several new CUSPs. The systems provides extended password security, sreater flexibility for the system manaser in controllins account usase, and sreater usase reportins. Currently implemented under V6A, LASER's features will be incorporated into V6B durins 1977. This paper will be presented by Michael Hall, who who has been a RSTS user since V3. Michael has for a lons time pushed for the improvement of CUSPs to a level expected in a timesharins environment. I sure this is a session you will want to attend this session which is scheduled for:

Wednesday June 1
1:30 - 3:00
Dalton Room

COMMUNICATIONS UNDER RSTS
SPRING DECUS

At the fall DECUS, I presented a paper on a "frontend" for an RSTS system. From the number of inquiries I've received, it is obvious that quite a few RSTS users are interested in alternative methods for doing communications with the timesharing system.

Surely, using DECNET is a fine approach; however, installation requirements or availability of software forces many of us along other paths.

I have reserved time at the spring DECUS for a panel discussion on data communications under RSTS. The intent is to have short presentations by panel members on their own approaches to communications, and then questions and discussion with the group.

I would appreciate hearing from users who have researched and/or implemented such systems and would participate in the discussion.

Stewart M. Shacter
The First National Bank of Saint Paul
332 Minnesota Street
St. Paul, Minnesota 55101
(612)-291-5304

"Energy Management Package"

Wyman-Gordon Company, Midwest Division is a PDP 11/45 RSTS/E User. We operate several manufacturing plants using large amounts of electricity and other fuels in the manufacturing process. Like all industry, we have a continuous energy conservation need.

Currently we are investigating the availability of a package to record (off-line) total plant energy consumption. Also the desired system would predict energy usage over a given combination of equipment (using a mix of fuels) and show differences in the use of alternate fuels or equipment.

We would like to hear from anyone who has this type package, doing similiar work or with the same needs.

DAVID W. JOZA
WYMAN-GORDON COMPANY
MIDWEST DIVISION
HARVEY, IL. 60426

Asynchronous Communications Systems

At the DECUS Symposium at BANFF I presented a paper on asynchronous communications systems with error-detecting protocol. The system I have developed is half-duplex and links a PDP-11 RSTS system with a PDP-10.I would like to set in touch with other users working on the same type on thing.

Patrick Power
% Systems Design And Devlopment
651 Glenayle Cres.
London, Ont.
N5X 1X8
Canada

RSTS/E Commercial Users Newsletter
April 1977

As was stated in the last issue, the MARKETPLACE section has been purged and items must be resubmitted to run in the next four (4) issues. So far, the response has been light. If your ad does not appear in this issue, and you wish it to, please resubmit.

With the recent release of 6B, I hope that we can solicit some business related articles concerning new techniques or improvements made possible by string arithmetic, NUM1$, etc. There are so many new things in 6B, that I'm sure each user won't try them all at once. We can all benefit more quickly if we can get these articles to the newsletter.

See you in Boston.

Murray Stinson
Scientific Placement, Inc.
2700 South Post Oak Road, Suite 2430
Houston, Texas 77056

===> EDITOR'S NOTE <===

After receiving this issue of the Marketplace, I was informed by the DECUS US Board that they are changing the policy on the inclusion of sofeware for sale in the SIG newsletter. There was not time to revise this Marketplace, but its revision will be discussed during the Commercial sub-SIG meeting at DECUS.

RSTS MARKETPLACE

DESCRIPTION	CONTACT
1.) Subroutines to phonetically code last names for storage and retrieval via standard phonetic techniques. 2.) Screen generator programs for the Tektronix 4023 buffered CRT terminal. 3.) Complete letter writing system using paragraph/letter files to build custom upper/lower case letters from form paragraphs. System allows insertions as well as a user composed paragraph.	Murray Stinson Scientific Placement, Inc. 2700 South Post Oak, #2430 Houston, Texas 77056 713-621-9811 4/1/77
1.) DATA BOSS/2 - Data Base Management Systems and Applications Monitor. Conversational setup, an English-like Inquiry Language, fast retrieval of millions of records via up to 65 keys. DBMS provides device independence and enables I/O with any ASCII file or device.	John H. Wright Florida Computer Inc. 99 N. W. 183 Street North Miami, Florida 33169 305-652-1710 4/1/77
"FIPS" - Financial Information Processing System. An easy to use computerized accounting system designed by accountants for accountants. Includes general ledger, payroll, payables, cost accounting, financial statements, etc.	Michael M. Niernberg Environmental Developers, Inc. 13693 East Iliff Avenue Denver, Colorado 80232 303-755-3377 4/1/77
The Statistical Programs for the Social Sciences/Carleton (SPSS/C) for the PDP-11. It is written in FORTRAN IV and tested under DOS V9, RT-11, and RSTS.	Graham Kimble or Carl Henry Computer Center Carleton College Northfield, Minnesota 55057 507-645-4431 Ext. 204,504 4/1/77

RSTS/E OMSI SORT-I

Fast SORT for Large Files. Machine language SORT Package compatible with DEC BASIC-Plus SORT runs up to 50 times faster than DEC version. Sorts files containing over one million records. Very attractive price with OEM and multiple CPU discounts. Runs on version 6B as normal 16k word user job without software or hardware modifications. Chain out and back or run stand-alone.

OMSI Computing
4015 S. W. Canyon Road
Portland, Oregon 97221
(503) 248-5900

RSTS Education Sub Sig Meeting

Hampton Room - Tues May 31

9:00 am

1. New RSTS Users in attendance

   A. Particular Interest areas or concerns, try to match with another in attendance

2. Report on Statistical packages available

   A. Michael Hall, Lawrence University, COSAP II

   B. Carl Henry, Carleton College, SPSS

   C. Any others that might be present

3. Attention to special sessions of interest

   A. Wise - brief comment on session

   B. CAI - brief comment on session

   C. System Security in Educational Environment

   D. Educational Product Group

   E. EDUSIG business meeting

   F. Quality programming at $1.96 per hour

4. Organization for Fall DECUS in San Diego

   A. Which sessions should be repeated, if any?

   B. What is the interest of the group for sessions - who will participate?

5. Howard has not provided commercial vendors with the Educational RSTS User address listing. Check with the group on this policy.

6. Educational RSTS USER Wish items, RSTS Concerns, etc.

   A. Price break for educational institutions on software maintenance update.

PAINTING with RSTS/E

The following program is sample program I worked up using the ECHO CONTROL option in V6B to "paint" fields on the screen of a VT52 and control the input of record information. Thanks are due Martin Minow of DEC for many of the ideas I used in the program.

The program is the first one to appear in the newsletter that makes use of the EXTEND mode under V6B. I hope some of you will take some time and send me some other programs to publish. You experienced users can do the newer RSTS users a real service by letting them see some of your ideas.

Bob Branton

PS: Programming Standard people forgive me!

```
10      !*************************************************************************
                    T P A I N T

        test screen painting program        unguarenteed by: R. Branton

15      N% = 1
20      OPEN 'KB:' AS FILE N%,MODE 8% \
               X9$ = SYS(CHR$(11%)+CHR$(N%))

50      EXTEND
100     !................ open and field output file ........... &

        OPEN 'DUMMY.FIL' AS FILE 2% \

               FIELD #2%,8% AS F2$(1%),25% AS F2$(2%),25% AS F2$(3%),
                     25% AS F2$(4%),2% AS F2$(5%),5% AS F2$(6%)

110       DUMRECORDCNT% = 0%

120            GOSUB 26100% !.....go set up screen  PAINT fields

2000    !.............clear screen and print titles ............. &

        PRINT #N%,FNA3$(1%,0%,FNA2$('J')) \

               FOR I% = 1% TO 6% \
               PRINT #N%,FNA1$(TROW%(I%),TCOL%(I%),TITLE$(I%)) \
               NEXT I%

2020    !................ set a customer record ............... &

               FOR P% = 1% TO 6% \
                 GOSUB 26000% !... go set next field           &
        \              GOTO 32000% IF P% = 1% AND A1$ = '0' !...stop &

               !==> any error checking for the field goes here &

        \        GOTO 2020% IF A1$ = 'G' !...goof -> restart customer &

        \      LSET F2$(P%) = A1$ !...set field to entered value &
        \      NEXT P%
```

DIGITCRAFT
P. O. Box 4249
Tucson, Arizona 85717


```
2040    !.............. is it to be recored ..................... &
        PRINT #N% FNA3$(TROW%(7%),TCOL%(7%),TITLE$(7%)) \

                P% = 7% \
                  GOSUB 26000% \
                    IF A1$ <> 'Y' THEN
                        PRINT #N%,FNA3$(1%,30%,'==> Not Recorded <==') \
                          SLEEP 2% \
                            GOTO 2000% !...so try again

2060            !................ Y  E  S ................... &

                DUMRECORDCNT% = DUMRECORDCNT% + 1% \

                        PUT #2%,RECORD DUMRECORDCNT% \

                FOR I% = 1% TO 5% \
                PRINT #N%,FNA3$(1%,30%+I%,'      Record ' +
                        NUM1$(DUMRECORDCNT%) + ' is recorded') \
                NEXT I% \
                        SLEEP 2% \
                GOTO 2000% !
        >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>><<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<

25999   !
        >>>>>>>>>>>>>>>>>> P A I N T routine <<<<<<<<<<<<<<<<<<<<<<<<<<

26000   PRINT #N%,FNA1$(XX%(P%),YY%(P%),STRING$(NN%(P%),PP%(P%))); \
        PRINT #N%,FNA$(XX%(P%),YY%(P%)); \
        PRINT #N%,RECORD 256%,CHR$(PAINTMODE%+NN%(P%))+CHR$(PP%(P%)); \

                GET #N% \
                        FIELD #N%,RECOUNT AS A$ \
                        A1$ = CVT$$(A$,4%) \
                          GETLEN% = LEN(A1$) \

        PRINT #N%,FNA1$(XX%(P%),YY%(P%)+GETLEN%,STRING$(NN%(P%)-GETLEN%,32%));

26090           RETURN !
        ..........................................................

26100   !.......... setup  P A I N T ............................. &

        DIM TROW%(25%),TCOL%(25%),TITLE$(25%),XX%(25%),YY%(25%),NN%(25%),PP%(25%)

        TROW%(I)-title row, TCOL%(I)-title col, TITLE$(I)-field title,
        XX%(I)-field row, YY%(I)-field col, NN%(I)-field length,
        PP%(I)-paint character

26120   PAINTNUM% = 7% !...number of paint fields
26140           FOR P% = 1% TO PAINTNUM% \
        READ TROW%(P%),TCOL%(P%),TITLE$(P%),XX%(P%),YY%(P%),NN%(P%),PP%(P%) \
                NEXT P%

26190           RETURN !
        >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>><<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
```

```
26200    DATA 4,10,'Enter customer #',4,25,8,95
26220    DATA 6,10,'Name:',6,16,25,95
26240    DATA 8,10,'Address:',8,19,25,95
26260    DATA 10,10,'City:',10,16,25,95
26280    DATA 12,10,'State:',12,17,2,95
26300    DATA 12,25,'Zip:',12,30,5,95
26320    DATA 20,40,'Record?',20,48,1,95
26499    !>>>
```

Converting V6A runtime systems to V6B

The following program will convert runtime system modules for version 6A to the proper format for version 6B.

Ross Miller
On Line Processing
N. 637 Hamilton
Spokane, Wa 99202

```
100 DIM S%(30)
1220 FIELD #2%, I% AS O$, 2% AS O$
10000 PRINT\INPUT "RUN-TIME SYSTEM NAME"; N$\ N$=CVT$$(N$,-1%)\
        I$="[0,1]"+N$+".RTS"
10100 OPEN I$ FOR INPUT AS FILE 1%
10110 CHANGE SYS(CHR$(6%)+CHR$(-10%)+I$) TO S%
10120 S%(1%)=6%\ S%(2%)=17%\ S%(3%)=255%\ S%(4%)=255%\
        CHANGE S% TO S$
10130 S1%=ASCII(MID(SYS(S$),13%,1%))
10140 FIELD #1%, 512% AS I$
10200 OPEN "[0,1]"+N$+".TMP" FOR OUTPUT AS FILE 2%, FILESIZE S1%+1%,
        MODE 16%
12000 S$=MID(SYS(CHR$(6%)+CHR$(-10%)+N$),7%,4%)
12010 S2%=(128%-S1%)/4%*2048%
12020 S$=S$+CVT%$(0%)+CVT%$(0%) ! BLANK IDENT
12030 S$=S$+CVT%$(SWAP%(1%))  !START BLOCK OF SAVE IMAGE
12040 S$=S$+CVT%$(SWAP%(S1%+1%))!START BLOCK OF SYMBOL TABLE
12050 S$=S$+CVT%$(0%)    !NUMBER OF GLOBAL SYMBOLS IN TABLE
12060 S$=S$+CVT%$(SWAP%(S2%))    !LOAD ADDRESS
12070 S$=S$+CVT%$(SWAP%(-S2%-2%))!MODULE SIZE
12080 S$=S$+CVT%$(SWAP%(1%))     !TRANSFER ADDRESS
12090 S$=CVT%$(SWAP%(1%))+S$     !COUNT OF MODULES IN SIL
12100 FIELL #2%, 22% AS O$\ LSET O$=S$\
        S%=0%                    !STORE INDEX ENTRY.  CLEAR CHECKSUM.
12110 FOR I%=0% TO 20% STEP 2%
12120 FIELD #2%, I% AS O$, 2% AS O$
12130 S%=S% XOR SWAP%(CVT$%(O$))
12140 NEXT I%
12160 FIELD #2%, 22% AS O$, 506%-22% AS O$\ LSET O$=STRING$(506%-22%,0%)
12180 FIELD #2%, 506% AS O$, 2% AS O$\ LSET O$=CVT%$(SWAP%(S%))
12190 FIELD #2%, 510% AS O$, 2% AS O$
12200 LSET O$=MID(SYS(CHR$(6%)+CHR$(-10%)+"SIL "),7%,2%)
12210 FIELD #2%, 512% AS O$
12220 PUT #2%
13000 FOR I%=1% TO S1%
13010 GET #1%
13020 LSET O$=I$\ PUT #2%
13030 NEXT I%
13040 CLOSE 1%, 2%
14000 ON ERROR GOTO 14010\ KILL "[0,1]"+N$+".RTS"\GOTO 14020
14010 RESUME 14020
14020 ON ERROR GOTO 0
14030 NAME "[0,1]"+N$+".TMP" AS N$+".RTS"
32767 END
```

DECAL

At STI we are very committed to DECAL. The Radio/TV department is basing their entire modular training program on it - both as a Job Control (CMI) technique and as a standard CAI teach/quiz environment. Consequently, I have had a great deal of correspondence with Tony DiCenzo, DEC Product Manager for DECAL. It appears there may be a completely redesigned, rewritten DECAL package within the next 18 months. Tony is looking for user input now ASAP. I have enclosed my comments and would ask other DECAL users to do the same.

Jon Perkins
Computer Centre
Saskatchewan Technical Institute
Sask. St. & Sixth Ave. N. W.
Moose Jaw, S6H4R4
Canada

1) there must be a way to convert present DECAL lessons to the new DECAL. Re-entering all lessons is not an acceptable alternative.

2) allow a method of letting <u>one</u> student re-do a lesson.

3) all programs in the DECAL system should accept answers to questions in upper-case, lower-case, or mixed upper- and lower-case

4) QUIZ (Lesson Administration Program) should allow answers to be in upper-case, lower-case, or mixed upper- and lower-case, unless the instructor specifies otherwise. For example, Remedial English lessons can deal with proper capitalization, as could Typing lessons. All other types of lessons would want upper-case and lower-case to be equivalent. The suggestion would be to have CREATE and LEDIT ask whether upper-case and lower-case should be equivalent at the start of each lesson.

5) Blanks should not matter when QUIZ checks the accuracy of an answer. For example, an anwer of 37.5 followed by a blank when entered as the answer during CREATE or LEDIT currently must be answered with an answer of 37.5 followed by a blank. Similarly, a name centered currently with a blank at the end to STUDNT must be entered with a blank at the end to QUIZ. This present situation is silly.

6) LREP should check whether a lesson exists (i.e. - if the .TST file exists) before looking for statistics on that lesson. This would provide an inexpensive check for an incorrectly spelled lesson name.

7) LREP should combine all answers to a given lesson by a single student into one reported entry in the following manner:

   - if a question is answered more than once as a result of branching the student back to the segment after he has already completed the segment, only the first response will be counted for that question

   In other words, LREP should report (and use for the derivation of statistics and the histogram) as a score for a student the number of questions successfully answered on the first attempt out of the number of unique questions asked the student.

8) To save the large amount of the computer's memory used when many QUIZ users are running simultaneoulsy, make QUIZ, at least, a run-time system so that only one copy of the program need be in memory at a time.

9) Provide QZ (quick entry, quick exit Job Control) as an alternative to QUIZ for those users in a short duration per person uses of DECAL.

10) Provide a "random branching" instruction or option in DECAL lessons. Might I suggest addition of the instructions BQR, BMR, and BTR (BQ random, BM random, and BT random). Each instruction would have two or more arguments:

```
BQR,5, 11,20
BMR,5,11,20,2
BTR,5,11,20
```

where the first three numbers mean:

Choose 5 questions at random (without replacement - -

do not administer the same question twice)

from segments 11 through 20 (inclusive).

This feature would solve the problem of cheating for CAI use, Job Control, and exam use. With this feature computer-administered exams would be a possibility.

11) BF (Branch to File) would be a lot more useful if there were a "RETURN" ("go back where you came from") instruction in the lesson/file you branched to. This would allow the same remedial work to be accessed from various lessons.

12) Commands used in lessons (BQ,A,BM, etc.) should allow a space or a comma to be be used to separate arguments.

13) When creating a lesson, the instructor should have the option of adding a weight to each question to effect the statistics and student scores produced by LREP. By making this a question at the start of creating a lesson, the instructor not interested in question weights will not have to be bothered answering the question "What weight do you wish to assign this question?" in each segment when an A command is entered.

Another, less foolproof, way of doing weighing would be to change the format of of the A command:

```
A,2,5
```

Where the 5 indicates the question has a weight of 5. If no weight is given, a weight of 1 is assumed.

This feature would allow complex questions to be given more credit than simple questions, make computer-generated exams even more feasible, and allow the instructor to give no credit for review questions - through a weight of zero.

14) Adding the concepts of random choice and weighted questions, invent another set of instructions BQWR, BMWR, BTWR that would look identical to BQR, BMR, and BTR, except the first number would now mean "Randomly choose any number of questions from segments 11 through 20 inclusive having a combined weight of 5".

This feature would make computer-administered exams a reality and one we could get most of STI using.

15) The new LEDIT (Lesson Editor) should use the same instruction format as the new PDP 11 general text editor.

16) Provide a convenient way of converting to and from DECAL source, so that DECAL lessons could be freely traded with other institutes (in source form).

O D D S A N D E N D S

from: Gary Oppenheimer
Fits Systems
One Red Oak Lane
White Plains, NY 10604

A. Have you ever needed a sense switch for your code? You can use the address switches in the front of the CPU panel (sorry 11/34 owners) to accomplish this.

```
10 X% = PEEK(-136%)
```

will move to X%, the value of the switches. As you usually run RSTS/E with all the switches up, X% is usually -1%. To test a certain condition, use the AND operation. For instance, to determine if bit 5 is down (remember: up is 1 , down is 0) use:

```
10 X% = PEEK(-136%) AND 16%  ! 2^5 = 16
```

If X% = 0, switch 5 is down. If X% = 16, its up.

One application has been to allow a given program to run in one of the two modes normal and test. If a given switch is down, the program lists specified variables on the terminal:

```
10 PRINT A$,B%,BUFSIZ(3%) IF (PEEK(-136%) AND 16%) = 0%
```

. . . . . .

B. A function to change 051176 to 11 MAY 76 (V$ = '051176'):

```
100         DEF FND4$(V$) = MID(V$,3%,2%)+'  '+
                    MID('JANFEBMARAPRMAYJUNJULAUGSEPOCTNOVDEC',
                          (VAL(LEFT(V$,2%))-1%)*3%+1%,3%)+'  '+
                       RIGHT(V$,5%)
```

. . . . . .

C. A function to convert RSTS Julian date to day of the week (V%=PEEK%(512%)):

```
100         DEF FND$(V%) \
               V1% = V%/1000%+70% \
               V%  = V1%/365%+(V1%+3%)/4% + V% -((V1%-70%)*1000%)-1% \
               V1% = V%/7% \
               V1% = V% - (V1% *7%)+1%
110          FND$ = MID('SUNMONTUEWEDTHUFRISAT',((V1%-1%)*3%+1%,3%)
```

. . . . . .

D. Convert MMDDYY format to RSTS Julian format (V$='MMDDYY'):

```
100         DEF FND1%(V$) \
            V7% = VAL(LEFT(V$,2%)) \
            V8% = VAL(MID(V$,3%,2%)) \
            V2%,V0% = VAL(RIGHT(V$,5%))
110               V4% = V0%-1% \
              IF V7% > 2% THEN V7% = V7%-3% ELSE
                  V7% = V7%+9% \
                  V0% = V0%-1%
120               V6 = (V7%*153.+2.)/5.+V8% \
            V7 = (V0%/4%)*1461.* FNM%(V0%,4%)*365. \
            V6 = V6 + V7 \
            V7 = V4%/4% * 1461. \
            V8 = FNM%(V4%,4%)* 1000% + INT(V6 - (V7+V8+306.)) \
   FNEND
130         DEF FNM%(V5%,V6%) = V5% - (V5%/V6%)*V6% ! integer modulo function
```

E. When using the RSTS/E ERRDIS program, it is often helpful to have a binary as well as octal dump output of the device registers. This makes comparing the ERRDIS output, and the documentation in the Peripherals Handbook easier. This is accomplished by 'OLD'ing in '$ERRDIS.SPL' from the distribution medium tape) and inserting lines 8110 and 8150 shown below. Compile the code under account [1,2], and save the source code. I suggest that you not alter the code on the distribution.

```
8110     PRINT #2%,C$(C%,I%-5%);':';TAB(10%);FNB$(E%(P%,I%));TAB(20%);FNB9$(E%(P%,I%))
         FOR I%=N1% TO N2%
8150     DEF FNB9$(U%)\
         U$=''\
         U$=U$+CHR$(48%+SGN(ABS(U% AND (2%**U1%))))
                 FOR U1%=15% TO 0% STEP -1%\
         U$=LEFT(U$,U1%)+' '+RIGHT(U$,U1%+1%) FOR U1%= 1% TO 17% STEP 4%\
         FNB9$=U$\
         FNEND

Ready

RUNNH

ERRDIS V06A-03
INPUT FILE (<CR> FOR DEFAULT)?
OUTPUT TO? KB:
OPTIONS? T4

OPTIONS: T4
   FILE: $ERRLOG.FIL
 OUTPUT: KB:
     AT: 15:47:09    , 08-DEC-76

TRAP THROUGH 4 ERROR AT 12:30:51    , 19-NOV-76
JOB#:      12
PSW:       170040
VIRT PC:   120734
ABS PC: 00140734
(PC-6):    011405     0 001 001 100 000 101
(PC-4):    016546     0 001 110 101 100 110
(PC-2):    000002     0 000 000 000 000 010

TRAP THROUGH 4 ERROR AT 10:51:31    , 06-DEC-76
JOB#:      13
PSW:       170051
VIRT PC:   113262
ABS PC: 00133262
(PC-6):    000757     0 000 000 111 101 111
(PC-4):    066503     0 110 110 101 000 011
(PC-2):    000002     0 000 000 000 000 010

TRAP THROUGH 4 ERRORS
        2

 2 ERRORS LISTED OUT OF 146 LOGGED SINCE 09:17:57    , 18-NOV-76
```

FROM THE TECHNICAL EDITOR #2

You will recall that in the last article we reviewed methods of doing input and output to simple and more complicated devices attached to the PDP 11. The data transfers were accomplished simply by loading the appropriate registers with information and then checking other status registers to determine what had happened to that information. The simplest example was a terminal which has a control and status register and a data register. I/O can be accomplished simply by putting data into the register then checking the control status register to make sure that the data has either been read or written.

It is possible to transfer data using interrupt structure available on the PDP 11. Using the control and status register, it is possible to set the interrupt enable bit which then causes a processor interrupt when data is either read or written. This allows the processor to go about other business without having to be constantly looking or polling the terminals to determine if there is input or output to be processed.

The next level of input/output also utilizes the interrupt facility, but in this case, more than one byte or word of data can be transferred at a time without involving the processor. This was the so-called cycle stealing mode. In this mode the processor steals a cycle or cycles from whatever programs happen to be operating and loads into the various registers information necessary to transfer one or more bytes of data; then returns to the program. When the data has either been successfully transferred or an error condition has arisen, an interrupt occurs from the device. The processor can then either steal more cycles in order to reload the registers to continue the transfer or go off to an error processing routine to process the error condition.

By way of an example, let us examine in more detail how a multiple data transfer might be accomplished from a specific location in memory onto magnetic tape utilizing the TU 10 magnetic tape drive. The PDP 11 Peripherals Handbook describes the operation beginning on page 4-501 with a description of the TM 11 controller and the TU 10 drive. The first register in the TM 11 is the status register. Each of the 16 bits in this register indicate the status of the particular drive in question. These include a bit for illegal command, end of file, cyclical redundancy check, parity error, bus grant late error, end of tape, record length error, bad tape error, non-

existent memory error, select tape, beginning of tape, whether it's a 7 or 9 channel tape, whether the tape is slowing down, whether the tape is write-locked, the current rewind status of the drive, whether the tape unit is ready. In addition to the status register, there is a command register. This 16 bit register is used to set information necessary for the data transfer. It is possible to set the density of the tape, to clear the tape drive, set the parity, select one of 8 units, enable the interrupt, set a particular function; that is, read or write, space forward or space backward. The final bit is a go bit telling the drive controller to begin the operation. A third register in the TM 11 is a byte record counter. This is simply a 16 bit binary counter which is used to count bytes in a read/ write operation. This register should initially be set by the program to the twos compliment of the number of bytes to be written on the tape. When this register becomes zero after the last byte of the record has been read or written, a control signal (interrupt) is sent indicating that there are no more data characters to be read or written from the tape. In addition to the number of bytes necessary to be transferred, a fourth register, the current memory address register, contains the memory address at which the transfer operation is to begin. Although this is a 16 bit register, it is possible to address 18 bits of memory. That is because in the command register bits 5 and 4 contain the two high order bits necessary for addressing the 18 bit address space. There are an additional two registers, a data buffer and read lines register, also in the TM 11.

In general, the following steps would be carried out in order to transfer data from memory to the TU 10 mag tape. First, using the command register, we would select the unit that we are planning to use. Selection is simply made by loading bits 10, 9 and 8 with the drive number we plan to use. After the unit has been selected, the status register is checked to find out the current status of the tape, and back once again to the command register to enable the interrupt. Since we want to write the data on the beginning of the tape, we should at this point rewind it. Using the command register, bits 3, 2 and 1 are function bits. Setting these bits all to 1 indicates that the function we wish the tape to execute is the rewind function. Finally, by setting bit 0 in the command register to 1, the tape will begin to rewind. It is not necessary for the CPU to wait while the tape rewinds since when the tape is rewound (we have enabled the interrupt), the TM 11 will interrupt the CPU and the status register will indicate that the tape has been rewound. Now, seeing that the tape is ready by looking in the status register, we would set the density and parity to the desired numbers, load the number of bytes that we wish to transfer

into the byte record counter, set the memory address at which we wish this transfer to begin in the memory address register, including the two bits that are contained elsewhere, if necessary, set the function bits to 0, 1 and 0 for bits 3, 2 and 1, respectively, indicating a write operation. Then, finally, set the go bit to 1 and the function will begin. Once again the processor goes off to do other functions. When the TM 11 is finished, it will interrupt the processor. If the data we wish to transfer is not in contiguous memory locations, when the current transfer is finished, the TM 11 interrupts the processor which then steals a cycle to load the next memory address into the memory address register, the number of bytes to be transferred into the byte record counter, sets the go bit and then returns from the interrupt. Thus, many data transfers can be made from different memory locations by the TM 11 utilizing only a few cycles of the CPU to effect these transfers.

What happens internally during this transfer of data from memory to the TU 10. Over what paths does the data flow. Over what paths do the control signals flow. How does the interrupt get back to the processor. These paths can vary depending upon the peripheral device and the PDP 11 model in question. Continuing to utilize our TU 10 example, the TU 10 is connected to the unibus of the PDP 11. Also connected to the unibus are the processor, memory, terminals, disks and many other types of input and output equipment. All data that flows from memory to the TU 10 flows via the unibus. This includes control signals and interrupt signals. Communication between the two devices on the unibus can be controlled by those two devices themselves. When the TM 11 is controlling transfer between the TU 10 and memory, it is in control of the unibus. Since the unibus is used by the processor in many I/O devices, a priority structure is used to determine which device can obtain control of the bus at any one time. By now we should know that all devices attached to the unibus have an address, (maybe more than one address), an interrupt vector, as described in the first Newsletter article, and a unibus priority which is known as bus request level.

The maximum transfer rate on the unibus using optimum device design would be one 16-bit word every 400 nanoseconds or 2.5 million 16-bit words per second. Although at first glance this seems very rapid, remember that the PDP 11 processors are capable of executing instructions in close to 1 microsecond. Assuming that each instruction that the processor executes has to be gotten from memory, as would be the case in a program, each of these instructions took 1 microsecond to execute. One out of every three words

DIGITGRAPH
P. O. Box 4249
Tucson, Arizona 85717

transferred via the unibus would consist of the processor getting its instructions from memory, leaving only 66% of the unibus available for transfer. The word "only" here means approximately 1.6 million words per second are transferrable via the unibus. Of course, the device is never optimum and considering that one RP04 is capable of transferring close to 400,000 words per second, it is possible to see how the bus could become the limiting factor in data transfers on a heavily I/O bound PDP 11. Referring now to the PDP 1170 Processor Handbood, figure 1.1 on page 1-2 indicates a slightly different architecture available for this model of the PDP 11. In addition to the regular unibus peripherals, which in this case could include terminals and TU 10 tape drives, there are four high-speed controllers which interface themselves to the unibus. However, the only signals that pass over the unibus are the control signals and status and interrupt signals as described earlier. The actual transfer of data occurs via special high-speed I/O bus to the mass storage peripheral, such as an RP04 or a TU 16 tape and via another high-speed bus directly linked to a special memory bus through the cache memory of the PDP 1170. This relieves the unibus of the high-speed data transfers between memory and peripheral devices. In this case at least, data can be transferred simultaneously via the unibus and the high-speed controller bus, sometimes called a mass bus, to the high-speed peripherals.

Since we have reviewed earlier input and output operations via TU 10 and TM 11 controller, it is interesting to compare the similarities and differences between this tape drive and a TJU 16 controller and TU 16 tape which operate utilizing a high-speed controller attached to the separate I/O bus and high-speed bus directly to the memory. In the first control and status register, bit 13 is set by a parity error on the control bus although the control bus is not the high-speed bus going to the high-speed device itself. Parity errors occuring on that bus are indicated in another register. The second register is a word count register which operates exactly like the matching register in the TU 10. Also, the unibus address register, which really is a memory address register, operates in a similar function to the one in the TU 10, even to the degree that the high order two bits are contained in the control and status register; one in order to give 18 bit addressing capability. While spacing forward records or characters was controlled in the TU 10 in a different fashion, the TU 16 has a different frame count register which contains a twos compliment number of records to be spaced over, characters to be written or characters that have been read. There is a second control and status register. This register contains some additional information including bit 8 which is the mass I/O bus data

bus parity error. This is the error that gets set on a parity error indication on the high-speed bus. The drive status register contains similar information, some additional peripheral information. One of the additional capabilities that the TU 16 has on an optional basis is to write in phase encoded 1600 bits per inch. There is an error register which each of the 16 bits contains a certain particular error. An intention summary register in the controller enables the programmer to determine immediately which of the 8 drives has interrupted rather than looking at the status of all 8 drives of the sequence. A character check register contains the cyclic redundancy check character that is used on the TU 16. There is the same data buffer register as well as a drive type register which allows the program to find out what type of drive this is. There are also serial number registers containing the serial number of the drive and a tape control register to control the density and format of the data being written. So, although the TU 16 has a few more registers than the TU 10, most of the additional data available is better error checking, increased error diagnostic aids and little, if any of it, has to do with the difference in architecture between the way the data is transferred via a high-speed controller. Thus, the fact that the TU 16 can operate via a mass bus and a high-speed controller is transparent to the programmer who has to program applications for it.

Carl B. Marbach
ACME-HARDESTY CO., INC.
910 Benjamin Fox Pavilion
Jenkintown, Pa. 19046
(215) 886-0200

Another SEND/RECEIVE

Although there are several Send/Receive routines floating around, I think you will find these the easiest to use. To send a message, use X$ = FNS$('message',A$,A%) where A$ is the Receiver's name in ASCII, or blank if A% is used. A% is the job number of the Receiver if A$ is unknown. If ASCII(X$) = 1 the Send worked. Otherwise, byte 1 is the Job number or the Sender, and bytes 2 to N is the data sent. At line 25130, change DIM statement to the Job Max of the system. At lines 25060 and 25220, the 'ON ERROR GOTO' should be set to the error routine in the application program.

Since NUM$ did not handle numbers longer than 6 digits in V6A, the functions FNN$ and FNF$ simulate NUM1$ and provide for leading zeros. The comments make the code self documenting.

Gary Oppenheimer
Fits System Inc.
One Red Oak Lane
White Plains, NY 10604

```
25010   !>>>
```

```
25080    DEF FNS1%(V3$,V4$)\
                 V5%=0%\
                 FNS1%=1%\
                 ON ERROR GOTO 25100
                          !CLEAR ERROR COUNTER
                          CLEAR ERROR FLAG
                          SET ERROR TRAP

25090    V2$=SYS(CHR$(6%)+CHR$(18%)+CHR$(-1%)+V4$+V3$)\
                 GOTO 25110
                          !SEND THE MESSAGE
25100    IF (ERR<>32% AND ERR<>5%) OR V5%>5% THEN
                 FNS1%=0%\RESUME 25110
                          !IF WE HAVE TRIED TO SEND MODE THEN 5 TIMES, OR
                           WE HAVE UNEXPECTED ERRORS, THEN RETURN THE ERROR
                          CODE,ABORT TRANSMISSION
25105    V5%=V5%+1%\
                 SLEEP 1%\
                 RESUME 25090
                          !THIS WILL HANDLE THE PROBLEM OF TRYING TO SEND
                          INFORMATION WHEN THERE ARE NO SMALL BUFFERS AVAILABLE
                          AND ONE IS NEEDED. WE WAIT FOR 1 SECOND AND TRY
                          AGAIN
25110    ON ERROR GOTO 32010\
                 FNEND
                          !WE ARE DONE
25120    !>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
         !       FUNCTION 'RECIEVER' FOR SEND RECIEVE
                 V5$-RECIEVERS NAME IN ASCII
                 DATA IN BYTE 1
                 ASCII(0)-BAD RECEPT WITH ERROR DATA IN NEXT 2 BYTES
                 ASCII(N)-JOB #N SENT DATA IN BYTES FOLLOWING
25130    DEF FNR$(V5$) \
                 DIM V5$(20%),V6%(20%),V3%(20%) \
                 V5$ = MID(SYS(CHR$(6%)+CHR$(-10%)+V5$),7%,4%) \
                 ON ERROR GOTO 25200
                          !DECLARE FUNCTION 'MESSAGE RECIEVE'
                          !GET RAD 50 NAME OF RECIEVER
25140    V6$=SYS(CHR$(6%)+CHR$(18%)+CHR$(2%)+CHR$(127%)+V5$)\
                 V5%=ASCII(RIGHT(V6$,5%))/2%\
                 V6%=ASCII(RIGHT(V6$,9%))\
                 IF V6% THEN 25160 ELSE V5$(V5%)=CHR$(V5%)+MID(V6$,13%,16%)
                                          !V5%-JOB #
                                          !V6%=# OF PACKETS
25145    V6%(V5%)=ASCII(RIGHT(V6$,10%))\
                 V3%(V5%)=CVT$%(RIGHT(V6$,11%))\
                 IF V6%(V5%) THEN 25140 ELSE 25180
                                          !V6%()-# OF PACKETS RECIEVED
                                          !V3%()-LEN OF MESSAGE RECIEVED
25150                    !GET A PACKET-GET THE SENDERS JOB NUMBER
                         GET PACKET NUMBER
                         IF PACKET #=0 ITS A HEADER
                         STORE PACKET DATA IN STACK
                         STORE TOTAL # OF PACKETS
                         IF 0 THEN NO MORE ELSE GET MORE
```

```
25160    IF V6%(V5%) <> V6% THENV5$(V5%) = CHR$(0%)+CHR$(12%)+CHR$(V5%)
                 ELSE V6%(V5%) = V6%(V5%)-1% \
                 V5$(V5%) = V5$(V5%)+MID(V6$,10%,19%) \
                 IF V6% <> 1% THEN 25140
25170                    !ARE WE OUT OF SEQUENCE???
                         IF WE ARE RETURN ERROR 12 (I/O ERROR) OTHERWISE,
                         DECREMENT PACKET COUNT
                         ADD DATA TO STACK
                         MORE PACKETS COMMING???
25180    FNR$=LEFT(V5$(V5%),V3%(V5%)+1%) \
                 V5$(V5%) = '' \
                 GOTO 25220
25190                    !RETURN THE ANSWER TO THE USER AND
                         CLEARE SOME CORE
25200    IF ERR=5% THEN RESUME 25140 ELSE
                 V5$(V5%) = CHR$(0%)+CHR$(ERR)+CHR$(ERL)+CHR$(V5%)+CVT%$(V3%(V5%))
                 RESUME 25180
25210                    !THE ONLY ERROR WE ALLOW IS FOR THE STANDARD
                         S/R ERRORS
25220    ON ERROR GOTO 32010\
                 FNEND
25855    !
         >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
                 FUNCTION 'EXTANDED NUMS'
                          CALLED AS X$=FNN$(A,B)
                                   WHERE A IS THE NUMBER TO TURN INTO A
                                         STRING
                                   WHERE B IS THE DESIRED LENGTH OF  THE STRING
                                   (LEADING ZEROS APPENDED IF NEEDED)
25860    DEF FNN$(V,V6%)\
         V9$=''\V%=-(V<0.)\V=ABS(V)\
         FOR V7%=18% TO 1% STEP -1%\
         V6=V/10**V7%\
         V9$=V9$+CVT$$(NUM$(ABS(FIX(V6*10%)-(FIX(V6)*10%))),2%)\
         NEXT V7%\
         FNN$=FNF$(V0%,STRING$(V%,45%)+RIGHT(V9$,19%-V6%+V%))\

         FNEND
25915    !
         !>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
                 FUNCTION 'ADD LEADING "0"'
                 CALLED AS A$=FNF$(A%,B$)
                          WHERE A% IS THE DESIRED LENGTH OF THE FULL NO.
                          AND B$ IS THE STRING NUMBER
25920    DEF FNF$(V4%,V5$)\
                 V5%=LEN(V5$)\
                 FNF$=STRING$(V4%-V5%,48%)+V5$\
                 FNEND
```

Nitty-Gritty RSTS/E Meeting

Let's Share Some Ideas

For System Programmers and Application Experts one of the most valuable opportunities is to get together with others to exchange ideas and experiences. Although DECUS meetings provide a forum for some of these types of exchanges, the diversity of the attendees and the large numbers involved, limit the depth of coverage for engrossing, important, but often unpopular (to the masses) topics.

Our plan is to set up a West Coast Nitty-Gritty Meeting for RSTS/E people whose expertise is established. We'll skip the easy topics and tackle the hard ones -- Writing Run-Time Systems, performance measurement, RSTS/E Bugs and Fixes, Disk performance, management utilities, heavily loaded systems, novel systems design, program optimization.

Interested? The tenative plan is for a Fri-Sat (Sunday optional) meeting some time in August to be held in a Buena Park or Anaheim, California hotel (that's near Disneyland and Knott's Berry Farm if you're interested). Data Processing Design will sponsor the meeting and provide computer access on Friday, and stand-alone time on Saturday and Sunday. Media transfer capability including 800/1600 BPI 9-track magtape, RK05, and floppy disks will be available.

The cost -- minimal.

If you are interested let us know. Tell us what topics you'd like to have discussed and what topics you can discuss as an expert; Write us a note or call us.

Richard A. Marino
Software Manager

Mike Mayfield
Senior Systems Analyst

Data Processing Design, Inc.
6980 Aragon Circle, Suite 8
Buena Park, CA 90620
(714) 994-4971